**PATENT APPLICATION**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of

Jurg PORTMANN Attn: PCT Branch

Application No. New U.S. National Stage of PCT/EP04/050776

Filed: December 5, 2005 Docket No.: 126067

For: METHOD AND TERMINAL FOR GENERATING UNIFORM DEVICE-INDEPENDENT GRAPHICAL USER INTERFACES

**SUBMISSION OF THE ANNEXES TO THE INTERNATIONAL PRELIMINARY REPORT ON PATENTABILITY**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Attached hereto are the annexes to the International Preliminary Report on Patentability (Form PCT/IPEA/409). The attached material replaces the material in the specification at page 2 and claims 16-20.

Respectfully submitted,

James A. Oliff
Registration No. 27,075

Eric D. Morehouse
Registration No. 38,565

JAO:EDM/crh

Date: December 5, 2005

**OLIFF & BERRIDGE, PLC**
**P.O. Box 19928**
**Alexandria, Virginia 22320**
**Telephone: (703) 836-6400**

DEPOSIT ACCOUNT USE AUTHORIZATION
Please grant any extension necessary for entry;
Charge any fee due to our Deposit Account No. 15-0461

design is well suited to display common parts of the same data on multiple screen masks. All controller layer application parts will act of the same model pattern such that all viewer layer application parts display the same data. In case of a field within the presentation of the data is modified the associated controller layer application part updates the model pattern such that the model layer application part notifies connected presentation elements provided by the viewer layer application part in order to effect updates presentation of the data.

It is self-explanatory that the layer design although structured into separate layers does not allow for independent implementation. Developers have to take care about consistency between the model layer application part and the connection of the viewer and controller layer application parts due to the involvement with each other. The maintaining of consistency counteracts the initial idea of a separate layered application design. That means, developers which deal with model pattern and model layer application part design, have to maintain in parallel aspects of the viewer layer application part and controller layer application part or to be more general aspects of the view and controller pattern.

A new approach to manipulating the components of a distributed Internet application with heterogeneous interfaces is described in the document "A Document-based Framework for Internet Application Control" by Todd D. Hodes and Randy H. Katz. Static XML-based documents describing required interfaces both on the client-side and on the service-side are parsed by the other party that adapts the data to the defined interface. One additional feature of solving the heterogeneous interface problem is the capability for a dynamic generation of user interfaces. Nevertheless, the disclosed framework still presents serious drawbacks in particular with respect to the description redundancy and heavy charge to be carried by the processing unit.

Another similar approach is disclosed in the document "UIML: an appliance-independent XML user interface language" by Marc Abrams et al. The XML-based language UIML (User Interface Markup Language) solves the problem of applications portability throughout a range of various appliances with very different hardware and processing capabilities (such as desktop computers, handheld devices, mobile phones, etc.). However, this solution still presents exactly the same drawbacks as the framework disclosed by the previously cited document.

The US patent 5,793,368 discloses a method and a system for displaying user interfaces and visual styles and dynamically switching between different visual styles. Users can modify or adapt the visual style of the user interfaces once displayed. A specific text-based language called UIL (User Interface Language) is used to define the user-specific elements of the user interface. The information is stored on a server and retrieved after the user's login. Once retrieved, the description is interpreted by the PGUI (Programmable Graphics User Interface) and displayed to the user. This solution suffers from the same problems, creating heavy charge to the processing unit any time a style has to be changed.

It is an object of the present invention to simplify the design and generation of user interfaces which include aspects of the viewer layer application part and controller layer application part. In particular, the user interface and to be precise a graphical user interface (GUI) is generated dynamically at runtime such that model layer application part and viewer / controller layer application part are clearly separated.

The object of the present invention is solved by a method for providing a screen mask of a user interface and a terminal device, which is adapted to perform this method.

According to an aspect of the invention for generating a user interface of a network node, whereas the user interface (GUI) is operable by a user to operate an application, the application is structured into a core application part responsible for handling data objects and a viewer/controller application part responsible for displaying said data and initiating actions on said data, wherein said viewer/controller application part is formed by said user interface, whereas a screen mask creating module for creating dynamically a screen mask of said user interface retrieves screen mask configuration data and widget configuration

16. Computer program product for establishing a user interface, wherein said computer program product comprises program code sections stored on a computer readable medium for carrying out the method of any one of the claims 1 to 13, when said computer program product is executed on a microprocessor-based device, processing device, a terminal device or a network device.

17. Terminal device adapted to establish a user interface, which is operable by a user to operate an application executed by said terminal device, which comprises a screen mask creating component (240) for creating dynamically a screen mask of said user interface (GUI), comprising:

a retrieval component (260, 270) for retrieving a screen mask configuration data (320) and widget configuration data (310), which comprises configuration data about at least one component (10 - 18; 410),

a parsing component (250, 230, 240) for parsing said screen mask configuration data (320) to obtain type information about said at least one component (10 - 18; 410) and to obtain individual settings of said at least one component (10 - 18; 410), and for parsing said widget configuration data (310) to obtain one or more component patterns (411, 412),

a widget creating component (230) for obtaining said at least one component (10 - 18; 410) on the basis of at least one component pattern (411, 412) corresponding to said type information and for applying said individual settings onto said at least one component (10 - 18; 410), and

a linking component (430) for linking said at least one component (10 - 18; 410) to at least one data object (460, 465).

18. Terminal device according to claim 17, comprising a component pattern repository (210) which caches at least one component pattern (411, 412) and from which at least one component (10 - 18; 410) is requested and an identification component (240) for identifying at least one component pattern (411, 412) corresponding to said extracted type information, wherein said widget creating component (240) is adapted to derive at least one component (10 - 18; 410) from said at least one identified component pattern (411, 412).

19. Terminal device according to claim 17, wherein said terminal device comprises as further components for initialization of said component pattern repository (210) a retrieval component (260, 270) for retrieving a component configuration (310), which comprises component configuration data about at least one component pattern (411, 412), and a parsing component (250, 230) for parsing said component configuration information, wherein said widget creating component (230) is adapted to create said at least one component pattern (411, 412) and to store said at least one created component pattern (411, 412) in said component pattern repository (210).

20. Terminal device according to claim 19, comprising a retrieval component (260, 270) for retrieving a component configuration (310), which comprises component configuration information about at least one component pattern (411, 412), an identification component (240) for identifying said component configuration information about said at least one component pattern (411, 412) corresponding to said extracted type information, and a parsing component (250, 230) for parsing said identified component configuration information, wherein said widget creating component (230) is adapted to create said at least one component pattern (411, 412) and to derive said at least one component (10 - 18, 510) from said at least one component pattern (411, 412).